draft-vasseur-mpls-backup-computation-02.txt
February 2003

Jean-Philippe Vasseur (Ed)

Anna Charny (Ed)

Francois Le Faucheur(Ed)
Cisco
Systems, Inc.
Javier
Achirica
Telefonica Data
Espagna
Jean-
Louis Leroux
France
Telecom

IETF Internet Draft
Expires: August, 2003

February, 2003

draft-vasseur-mpls-backup-computation-02.txt

MPLS Traffic Engineering Fast reroute: bypass tunnel path computation for bandwidth protection

Status of this Memo

material
or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at
http://www.ietf.org/ietf/1id-abstracts.txt.
The list of Internet-Draft Shadow Directories can be accessed at
http://www.ietf.org/shadow.html.

Content

Abstract

This draft proposes an efficient model called ''Facility based computation model'' for computing bypass tunnels paths in the context of the MPLS TE Fast Reroute, while allowing bandwidth sharing between bypass tunnels protecting independent resources. Both a centralized and a distributed path computation scenarios are described. The

required
signaling extensions are also addressed in the draft.

1. Terminology

LSR - Label Switch Router

LSP - An MPLS Label Switched Path

PCS - Path Computation Server (may be any kind of LSR (ABR, ...) or a centralized path computation server

PCC - Path Computation Client (any head-end LSR) requesting a path

computation of the Path Computation Server.

Local Repair - Techniques used to repair LSP tunnels quickly when a node or link along the LSPs path fails.

Protected LSP - An LSP is said to be protected at a given hop if it has one or multiple associated bypass tunnels originating at that hop.

Bypass Tunnel - An LSP that is used to protect a set of LSPs passing over a common facility.

PLR - Point of Local Repair. The head-end of a bypass tunnel.

MP - Merge Point. The LSR where bypass tunnels meet the protected LSP.

A MP may also be a PLR.

NHOP Bypass Tunnel - Next-Hop Bypass Tunnel. A bypass tunnel which bypasses a single link of the protected LSP.

NNHOP Bypass Tunnel - Next-Next-Hop Bypass Tunnel. A backup tunnel which bypasses a single node of the protected LSP.

Reroutable LSP - Any LSP for which the "Local protection desired"

bit is set in the Flag field of the SESSION_ATTRIBUTE object of its Path messages (and/or

a FAST-REROUTE object is included in its Path message).

CSPF - Constraint-based Shortest Path First.

2. Introduction

The focus of this document is ''Bandwidth protection'' in the context of local repair capability of MPLS Fast Reroute. We concentrate on the issues related to the computation of bypass tunnels satisfying capacity constraints. We do not propose another method for MPLS traffic Engineering Fast Reroute. This draft makes the assumption that the fast reroute technique named Facility backup and described in [FAST-REROUTE] is used to provide fast recovery in case of link/node failure.

The exact algorithms for placement of the bypass tunnels with bandwidth guarantees are outside the scope of this draft. Rather, we concentrate on the mechanisms enabling the bypass tunnel path computation to be performed by a server which holds sufficient information in order to achieve efficient sharing of bandwidth between bypass tunnels protecting independent failures. The mechanisms are described in the context of both a centralized (the server computes the set of bypass tunnels to protect every facility in the network) and a distributed computation (every LSR is a server to compute the set of bypass tunnels for each of its neighbors in case of its own failure/link failure).

We specifically address the signaling involved for such computation between the PLR and the server (also called PCC-PCS signaling).

3. Background and Motivation

As defined in [FAST-REROUTE], a TE LSP can explicitly request to be fast protected (in case of link/node failure the TE LSP will be

locally
rerouted onto a backup tunnel, as defined in [FAST REROUTE]) and
rerouted onto a backup tunnel with an equivalent bandwidth (in
other
words without QOS degradation, supposing here that offering an
equivalent QOS can be reduced to preserving bandwidth
requirement).
This can be signaled (in the Path message) in two ways:

- with the SESSION-ATTRIBUTE object by setting:
  - the ''Local protection desired'' bit
  - the ''Bandwidth protection desired'' bit
- with the FAST REROUTE object

Note that other parameters related to the backup tunnel can also
be
signaled in the Path message.

Bandwidth protection will typically be requested for TE LSPs
carrying
very sensitive traffic (Voice trunking, ...).

When a link or a node failure occurs, the PLR (Point of Local
Repair)
fast reroutes the protected LSPs onto their bypass tunnel. The
PLR may
also send a Path Error notifying the head-end LSRs that the
protected
LSPs have been locally repaired so that head-ends should trigger
a re-
optimization, and potentially reroute the TE LSP in a non
disruptive

fashion (make before break) following a more optimal path,
provided
such a path exists.

The bandwidth of the bypass tunnels that the protected LSPs will
be
rerouted onto will dictate the level of bandwidth protection and
so the
QOS during failure until the TE LSPs are being re-optimized (if
such a
re-optimization can be performed, depending on the available

network
resources).

Various constraints can be taken into account for the bypass
tunnels:
(1) must be diversely routed from the protected
element
(link/node/SRLG diverse),
(2) must be setup in such a way that they get enough
bandwidth so that the protected LSPs requesting
bandwidth
protection should receive the same level of QOS when
rerouted. Note that the notion of bandwidth
protection is on
a per LSP basis.

(1) must always be satisfied and makes FRR an efficient
protection
mechanism to reroute protected TE LSP in 10s of milliseconds in
case of
link or node failure.

(2) allows FRR to provide an equivalent level of QOS during
failure to
the TE LSPs that have requested bandwidth protection.

4. Various bypass tunnel path computation models

Various bypass tunnel path computation models have been
proposed:
independent CSPF-based computation, [KINI], [BP-PLACEMENT], ...
A new
model, named ''facility based computation model'' is proposed in
this
draft.

5. Limitations of the independent CSPF-based computation model

The simplest mechanism (called independent CSPF-based
computation
model) to get bandwidth protection available today is to rely on
existing IGP TE advertisement and for the head-end of the bypass
tunnel:
- to determine the bandwidth requirements of the desired
bypass
tunnel(s),
- to compute the bypass tunnels path in the network
where the
appropriate amount of bandwidth is available using

standard CSPF-based computation,
- to signal the bandwidth requirements of the individual bypass tunnels explicitly.

While this approach is quite attractive for its simplicity, it presents a substantial set of challenges:
- Inability to perform bandwidth sharing between bypass tunnels protecting independent resources,

- Potential inability to find a placement of the bypass tunnels satisfying the bandwidth constraints.

5.1. Bandwidth sharing between bypass tunnels

Since local repair is expected to be used for only a short period of time after failure, typically followed by re-optimization of the affected primary LSPs, it is reasonable to expect that the probability of multiple failures in this short period of time is small. As a result, being able to share bandwidth on the link by bypass tunnels protecting different failures typically results in large savings in the bandwidth required for protection. This is what we refer many times in this document as ''efficient bandwidth sharing'' or as achieving ''bandwidth sharing''. Note also that the single failure assumption needed for such bandwidth sharing is a pre-requisite to any protection approach which uses pre-computed protected paths, clearly even two completely link and node disjoint pre-computed paths can both fail if more than one failure can occur as on failure may occur on the primary and the other on the second path. It is worth underlining that the single failure of a SRLG may result in the actual failure of multiple links. For the purposes of this draft we consider the entire SRLG as a single element that needs to be protected.

Once the head-end receives the Path Error (''Tunnel locally repaired''), reoptimization should be triggered followed by an LSP reroute making use of the ''Make Before Break'' technique to avoid traffic disruption, assuming such a more optimal path obeying the constraints within the new network topology can be found. If such a path cannot be found, the TE LSP will not be reoptimized and will still be fast rerouted by the immediately upstream PLR attached to the failed element.

The two following situations result in a multiple independent failures scenario where bandwidth protection with backup bandwidth sharing cannot be ensured:

- a second failure occurs before the TE LSP is reoptimized,
- the TE LSP cannot be reoptimized and a second failure happens before the first failure has been restored.

Note however that in networks where bandwidth is a reasonably available resource, this situation is unlikely to happen as the TE LSP reoptimization will succeed. Furthermore, in networks where bandwidth is a very scarce resource, bandwidth protection without backup bandwidth sharing is likely to require substantially more bandwidth, and therefore is likely to be impossible anyway.

As a result, bandwidth sharing among bypass tunnels protecting independent failures is highly desirable.

Previous approaches to achieve such bandwidth sharing have been proposed in [KINI] and [BP-PLACEMENT]. In [BP-PLACEMENT], extensive routing extensions are proposed to propagate the set of bypass LSPs and their

attributes. While the approach described in [KINI] substantially reduces the amount of state that needs to be propagated in routing updates, it sacrifices the amount of achievable sharing.

Both approaches require modifications to admission control, as well as signaling extensions required to perform specific call admission control for backed-up LSPs.

In contrast, the approach described in this draft can be used to achieve complete sharing without any routing extensions and without any modification to admission control (although as discussed further in section 6.2 a small amount of routing extensions is desirable for the distributed case to provide flexibility in the choice of protection strategies)

5.2. Potential inability to find a placement of a set of bypass tunnels satisfying constraints

Another well-known issue with independent CSPF-based computation with explicitly signaled bandwidth requirements is its potential inability to find a placement of the bypass tunnels satisfying the bandwidth constraints, even if such a placement exists. This issue is not specific to the placement of the bypass tunnels - rather it is due to the sub-optimality of a greedy on-demand nature of the CSPF approach and the non coordinated bypass tunnel computation approach to

protect a
given facility

See appendix A for a detailed example.

While addressing this problem is not a primary goal of this draft, facility-based computation model described in this draft provides the opportunity to improve the chance of finding a feasible placement of the bypass tunnel as it enables the use of algorithms that can take advantage of coordination between the placement of bypass tunnels protecting the same element. However, the exact algorithms appropriate for this purpose are outside of the scope of this draft.

6. Facility based computation model

In this draft we propose another model for the bypass tunnel path computation referred as the ''Facility based computation model''.

The facility based computation model can be implemented in two different ways: centralized or distributed. In all of these scenarios the facility based computation enables efficient sharing of bandwidth among bypass tunnels protecting independent failures. In addition, all of these scenarios also allow overcoming some of the limitations of the greedy independent CSPF-based placement of the bypass tunnels, increasing the chances of finding a bypass tunnels placement satisfying the constraints if such a solution exists. While some of these

approaches can benefit from an IGP-TE extension advertising an additional backup bandwidth pool, all of these approaches can be usefully deployed in a limited fashion in the existing networks without

any additional routing extensions at all. As shown bellow, the required
signaling extensions could be based on [PATH-COMP] with one additional
object (described in section 11.).

Note that in this section we assume that a bypass LSP protects only one
element (link, node or SRLG). The facility based computation model can
be extended to more general case where bypass tunnel can protect more
than one element, but this requires specific procedures that are addressed in sections 7 (NNHOP activated in case of both link and node
failures) and 8 (NHOP protecting link belonging to multiple SRLGs).

6.1. Centralized backup path computation scenario

In the centralized scenario, the bypass tunnel path computation is
being performed on a central PCS (which can be a workstation or another
LSR). The PCS will be responsible for the computation of the bypass
tunnels for some or all the LSRs in the network. Typically, there could
be one PCS per area in the context of a multi-area network. The PCS(s)
address may be manually configured on every LSR or automatically discovered using IGP extensions (see [IGP-CAP] and [OSPF-TE-TLV]).

To compute the bypass tunnels protecting a given element, the server
needs to know:

- the network topology,
- the desired amount of primary traffic that needs to be bandwidth protected (this could be either the actual bandwidth reserved by primary TE LSPs requiring bandwidth protection or the bandwidth pool that could be reserved by the primary LSPs - see Appendix A for a detailed discussion),
- the amount of bandwidth available for the placement of the bypass tunnels (also referred to as backup bandwidth).

The network topology is available directly from the IGP TE database as well as the desired amount of primary traffic that needs to be protected if one protects a bandwidth pool (and not the actual bandwidth reserved by primary TE LSPs requiring bandwidth protection).

The information about the backup bandwidth pool depends on the exact model and is discussed separately in each case.

However, whether or not this information is sufficient, depends on whether the server is also responsible for the computation of primary tunnels or not. This is discussed below.

6.1.1. Server responsible for both the primary and bypass tunnels path computation

In this scenario, the PCS can easily take advantage of knowing all the primary tunnels to define bandwidth protection requirements based on actual primary LSPs.

There is substantial flexibility in choosing what bandwidth can be used for the bypass tunnel placement. One approach might be to use for the bypass tunnels the same bandwidth pool as the corresponding primary LSPs.

At some point the user will have to specify the policy to the server. For example, protect traffic of a pool X with a bypass tunnel in the same pool but also the proportion of pool X that can be used for backup and primary. For pool X, the user could specify: ''up to y% of

pool X
can be used for backup''.

Since in this scenario the server is responsible for the placement of both the primary traffic and the bypass tunnels, at any given time in the computation of the bypass tunnels it has complete information about the topology and the current placement of all bypass and primary tunnels. Therefore, the server can compute the bypass tunnels protecting one element at a time, and when placing its bypass tunnels simply ignore the bandwidth of any bypass tunnels already placed if those protect a different element, thus ensuring implicitly the desired bandwidth sharing. In this case, there is no need to specify a notion of backup bandwidth pool.

PCC-PCS signaling

Having computed the bypass tunnels, the server needs to inform the head ends of the bypass tunnels about the placement of the bypass tunnels, their bandwidth requirements, and the elements they protect.

Depending on whether the server is an LSR or not, this could be done either via a network management interface, or signaled using RSVP extensions similar to those described in draft [PATH-COMP] (with a new RSVP object needed to achieve this communication described in section 11).

If the path computation server uses a network management interface to obtain the topology information and communicate the paths of the computed bypass tunnels to their head ends, this approach requires no signaling extensions at all. However, in the case when the path computation server is an LSR itself, additional signaling mechanisms are required to communicate to the server a request to compute bypass tunnels for a particular element, and for the server to communicate the bypass tunnels and their respective attributes to their head-ends. These extensions, described in detail in sections 11 are built on those proposed in [PATH-COMP]. Of course, the same extensions could be also used even if the PCS is a network management station.

Note that the benefit of having an LSR be the PCS as opposed to an off-line tool is the LSR's real-time visibility to any topology changes in

the network (unless the off-line PCS participates to the routing domain). In particular, the LSR-based approach can be expected to recompute the bypass tunnels affected by a failure much faster than a network-management based solution, thus making a single failure assumption more reliable. In addition, as will be discussed later in section 6.2, the ability of an LSR to compute bypass tunnels for other elements is especially useful in the context of a more distributed bypass tunnel computation.

Signaling Bypass tunnels with zero Bandwidth

Once an LSR has received the information about the bypass tunnels for one or more elements it is the head-end for, it needs to establish those tunnels along the specified paths. At first glance, given the need to ensure bandwidth protection, it seems natural to signal the bandwidth requirements of the bypass tunnel explicitly. However, as discussed in [BP-PLACEMENT], such approach requires that the local admission control is changed to be aware of the bandwidth sharing, and additional signaling extensions need to be implemented to enable an LSR to tell a primary LSP from a bypass LSP so that admission control can be performed differently in the two cases.

However, since the placement of both the primary and the bypass tunnels in this case is done by the server which maintains the bandwidth requirements of all these primary and bypass LSPs, it is sufficient to signal zero-bandwidth tunnels, thus avoiding the need for any additional signaling extensions or changes to admission control. Even though the required bandwidth will not be explicitly signaled, it will nevertheless be available along the path upon failure by virtue of the computation of this placement by the server which is fully aware of the global topology and places all TE LSPs in such a way that their bandwidth requirements are satisfied.

Note also that although the bandwidth requirements are not explicitly signaled, the head-end may store this information locally, since it may be needed in determination of which primary LSPs to assign to which bypass tunnels in the case where more than one bypass tunnel exists (see section 14).

6.1.2. Server responsible for bypass tunnels path computation only (not primary TE LSPs)

The main benefit of the previous scenario (PCS computing both the primary and backup LSPs) was due to the fact that the PCS could make use, for the bypass tunnels, of any available bandwidth not reserved for primary TE LSPs. As a consequence, this was not requiring a separate backup pool. On the other hand, if the PCS is just responsible for the bypass tunnels paths (i.e the primary tunnels are established on-line or by any other mechanism external to the backup path computation server), and if the bypass tunnels are signaled with

zero
bandwidth to enable efficient bandwidth sharing, then the bypass

tunnels cannot draw bandwidth from the same pool as the primary traffic they protect. This is because the bandwidth used by the bypass tunnels is invisible to the entity responsible for the computation of the primary TE LSPs and therefore the primary TE LSPS could make use of the entire bandwidth of a given pool. Therefore if the PCS used for bypass tunnel path computation uses any bandwidth of the same pool bandwidth protection violation might occur. Achieving efficient bandwidth sharing in this case requires the definition of a separate pool that could only be used for bypass tunnels. We refer to this pool as a backup pool.

Note that the notion of backup bandwidth pool is similar to that described in [BP-PLACEMENT].

The backup bandwidth pool approach can be used in two ways:
- being advertised in IGP
- without being advertised in IGP

Backup Pool advertised in IGP

In this approach, an additional bandwidth pool is established, and is flooded in the routing updates. See section 10 for more details.

If the backup path computation server uses the value of the backup bandwidth pool for its computation, no bandwidth overbooking will ever occur, since the primary tunnels now use the bandwidth from a different pool. The additional state that needs to be flooded in routing updates to implement the backup bandwidth pool does not impact the IGP scalability as the bandwidth protection pool being announced by IGP-TE is a static value, it does not dynamically change as backup TE LSP are set up, which preserves IGP scalability. As the bandwidth protection pool is being defined on a per link basis, this allows for different policies depending on the link characteristics.

Backup Pool not being advertised in IGP

The routing extensions discussed in the previous section are desirable but not necessary to deploy this approach in the existing network in a limited, but nevertheless useful fashion.

Since the computation of the bypass tunnels in this approach is performed by a centralized server, the server can use the notion of the backup bandwidth pool implicitly. Just as in the case of a server computing the placement of both primary and backup LSPs, such policy may be simply configured on the server for every link. The policy must ensure that the backup pool never overlaps with the pool requiring bandwidth protection.

A generic approach could be for the PCS to compute, for each link, the backup bandwidth as: link-bandwidth - maximum reservable bandwidth.

This approach requires that link-bandwidth > maximum reservable bandwidth which prevents the user from allowing TE overbooking.

Another approach could be manually specifying on the PCS for each link the backup bandwidth pool size. A separate policy can be configured for each link, depending for instance on their link speed.

Thus, substantial benefits may be achieved in this approach without actually deploying any additional IGP-TE extensions at all. The only drawback is that the policy will have to be the same for the whole network or may be specified on a per link basis which requires some extra configuration work on the PCS.

As in the previous approach (section 6.1.1)
- Signaling extensions can be used between a PCC and a PCS whether the PCS is an LSR or a network management station,
- Bypass tunnels are signaled with zero bandwidth.

6.2. Distributed bypass tunnel path computation scenario

While there are several clear advantages of a centralized (off-line) model, there are also well-known disadvantages of it (such as the single point of failure, the necessity to provide reliable communication channels to the server, etc.) While most of these issues can be addressed by the proper architectural design of the network, a dynamic distributed solution is clearly desirable.

This section presents the use of the ''facility-based computation'' solution in a distributed bypass path computation scenario.

6.2.1. Node Protection

Consider first the problem of node protection. The key idea is to shift the computation of the bypass tunnels from the head-ends of those bypass tunnels to the node that is being protected. Essentially, each node protects itself by computing the placement of all the bypass tunnels that are required to protect the bandwidth of traffic traversing this node in the case of its failure. Once the bypass tunnels are computed, they need to be communicated to their head-ends

(in this case the neighbors of the protected node) for installation.
The bypass tunnel head-ends play the role of PLR. Essentially, each node becomes a PCS for all of its neighbors, computing all NNHOP bypass tunnels between each pair of its neighbors which are necessary for its own protection. The fact that the bypass tunnels to protect a node X are being computed by a single PCS (node X) is essential and much more efficient than the non-coordinated independent CSPF-based computation.

The key pieces that make this model work are those already described in the context of the centralized server:

1) Making use of explicitly defined backup bandwidth pool which is logically disjoint from the primary bandwidth pool,

2) Taking advantage of a single failure assumption to achieve bandwidth sharing,
3) Installing bypass tunnels with zero bandwidth.

These three things together allow the computation of the placement of bypass tunnels for a given node to be completely independent of the placement of bypass tunnels for any other node. Essentially, each node has the entire backup bandwidth pool available for itself. The problem it needs to solve is how to place a set of NNHOP bypass tunnels (one or more for each pair of its direct neighbors) in a network with available capacity on each link equal to the backup bandwidth pool. This problem can be solved by any algorithm for finding a feasible placement

of a
set of flows with given demands in a network with links of given capacity.

While the details of such algorithm are beyond the scope of this draft,
it is clear that since the node now has control over all bypass tunnels
protecting itself, it is more likely that it can find such a placement,
and potentially find a more optimal placement, than is possible if the
head-ends of the bypass tunnels compute the placement of these tunnels
independently of each other.

Just as in the case of a centralized server, installing the bypass
tunnels with zero bandwidth ensures that no changes to admission control are necessary to allow sharing of the backup pool by bypass
tunnels protecting different nodes, thus enabling bandwidth sharing
between independent failures. Yet, by virtue of the computation, the
bypass tunnels protecting a given node will also have enough bandwidth
in the case of that node's failure.

Note also that the backup pools can be implicitly derived from the
routing information already available. This could be done by configuring max global reservable pool to being less than the link
speed by the desired value of the backup pool. Every node computing its
bypass tunnels then can by default use link speed minus the max global
reservable pool as the value of the backup pool to use in its computation of the bypass tunnels placement.

As described earlier, there is substantial benefit in defining the
backup pool explicitly and advertise its value as part of the topology
in the routing updates. This clearly requires an IGP-TE extension as
described in section 10. The benefit of doing so is that it provides
much more flexibility in the design of the network.

Yet it is important to emphasize that while IGP-TE extensions is a clear benefit for facility-based computation, it is not a requirement for this solution to work under a limited set of assumptions (namely, as discussed above if the backup pool is set to link speed minus maximum reservable primary bandwidth, the latter being configured to less than link speed).

Finally, signaling extensions required for communication between the node serving as path computation server and the head-ends of the bypass tunnels are the same as for an off-line server and are defined in sections 10.

6.2.2. Link Protection

In order to protect a link with MPLS TE Fast Reroute in both directions, two bypass tunnels protecting each direction of this link are installed by the corresponding head-end of that link. To make sure that traffic requesting bandwidth protection traversing this link is protected in case of a link failure (if both directions fail simultaneously), it is necessary to account for the interaction of the bypass tunnels protecting different directions of this link. That is, one needs to make sure that if a bypass tunnel T1 protecting bandwidth B1 on a directed link A->B and the tunnel T2 protecting bandwidth B2 on a directed link B->A traverse the same directed link L, then link L has spare capacity of at least B1+B2.

If the two ends of the link compute their bypass tunnels independently, the way to ensure this condition would be to explicitly signal the bandwidth of the bypass tunnels. However, as discussed earlier, this approach makes the sharing of bandwidth between the bypass tunnels protecting different elements impractical and would require IGP and admission control extensions. To achieve this goal in a distributed setting we propose that one of the two end-nodes of the link takes the responsibility for computing the bypass tunnels for both directions using the backup pools explicitly or implicitly defined. We propose that by default the node with the smaller IGP id serves as the server (PCS) for the other end of the link. Therefore, by default a node with id X serves as a PCS for NNHOP bypass tunnels protecting itself and NHOP bypass tunnels protecting any adjacent bi-directional link for which the other end has an IGP id larger than X.

6.2.3. SRLG protection

In the case when each link in the network cannot belong to more than one SRLG, we propose to use exactly the same approach as for the bi-directional link. That is, if an SRLG consists of a set of bi-directional links, the node with the smallest IGP id of all the endpoints of these links serves by default as a path computation server. The case where links are part of more than one SRLG requires specific processing (see section 8).

6.3. Signaled parameters

The PCC (an LSR) will send a bypass tunnel path computation request to the PCS using the RSVP TE extensions defined in [PATH-COMP] and the newly BACKUP-TUNNEL object defined in this draft.

The PCC's request will be characterized by the specification of several parameters that are discussed bellow.

6.3.1. Element to protect

The PCC specifies the element to protect: Link, Node or SRLG. Typically, a link protection request will result in a set of NHOP bypass tunnels as a node protection request will result in a set of NNHOP bypass tunnels.

6.3.2. Bandwidth to protect

There are two different approaches for the bandwidth to protect constraint:

- The bypass tunnel bandwidth may be based on the amount of reservable bandwidth pool on a particular network resource,
- The bypass tunnel bandwidth may be based on the sum of bandwidths actually reserved by established TE LSPs requiring bandwidth protection on a particular resource.

Each approach is having pros and cons that are being extensively discussed in Appendix B.

6.3.3. Affinities

The requesting node may also specify affinities constraint. Affinities for the bypass tunnel may be configured on the PLR by the network administrator or derived from the FAST-REROUTE object of the protected TE LSP, if used. In this former case, this would require some rules to derive the affinities of the bypass tunnel from the affinities of the protected TE LSPs making use of this bypass tunnel.

6.3.4. Maximum number of bypass tunnels

It may happen that no single bypass tunnel can fulfill the constraints requirements. In such a situation, a set of bypass tunnels could be computed such that the sum of the bandwidths of every element in the set is at least equal to the required bandwidth. It may be desirable to bound the number of elements in this set by specifying a maximum number of bypass tunnels originating at a PLR and protecting an element.

6.3.5. Minimum bandwidth on any element of a set of bypass tunnels

When a solution can be found with a set of bypass tunnels it may also be desirable to provide some constraint on the minimal bandwidth value for any bypass tunnel in the set. As an example, if a 100M bypass

tunnel is required, a set of 1000 tunnels each having 100K is likely to be unacceptable. Also, it is worth reminding that a single protected TE LSP will make use of a single bypass tunnel at a given time.

6.3.6. Class Type to protect

Specifies the Class-Type(s) to protect. See section 8 on operations with DS-TE.

6.3.7. Set of already in place bypass tunnels

In certain circumstances (stateless PCS), it may also be useful for the PCC to provide to the PCS the set of already in place bypass tunnels with their corresponding constraints for the PCS to try to minimize the incremental changes of the existing bypass tunnels due to the placement of new bypass tunnels.

7. Validity of the Independent failure assumption

The facility based computation model is heavily dependent on the single independent failure assumption. That is, it is assumed that the probability of multiple independent element failures in the interval of time required for the network to re-optimize primary tunnels affected by a given failure and to re-compute the bypass tunnels for other elements is low.

In a distributed model both of these tasks are likely to be accomplished within a very short time so the assumption typically can be justified. The loss of bandwidth protection in the rare cases that the assumption is violated is offset by the benefit of sharing the bandwidth between bypass tunnels protecting different elements.

However, not all elements are independent. One example of elements that are not independent is a set of links in the same SRLG. Therefore, as discussed above, SRLG is treated as a single element and is protected as a single entity.

Another example of failures that are not independent is a failure of a node and links adjacent to it. It is possible (and is frequently the case) that a failure of a node results also in the failure of the link(s). However, in the approach described in the draft the computation of bypass tunnel paths for link and node protection

is done
independently. This is necessary to ensure that NNHOP tunnels
for a
node can be computed completely independently of the NHOP
tunnels for
adjacent links, thus enabling the distributed computation. The
justification for this is that when a node fails, traffic that
does not
terminate at this node is protected because it is rerouted over
the
NNHOP tunnels, and traffic that does terminate at the failed
node does

not need to be protected against the failure of adjacent links
since it
is dropped anyway.

Thus, the underlying assumption is that if a node fails, the
NHOP
tunnels protecting the link are not used, while if a link fails
but the
router does not, the NHOP tunnels are used. So they can in fact
be
computed independently. However, this reasoning only works if it
is in
fact possible to identify the type of failure correctly and use
the
appropriate set of tunnels depending on the failure.

There are several cases to be considered:

- A downstream router fails but the link does not,
- The link fails but the downstream router does not,
- The link fails because the downstream router failed.

The first case is typically identifiable by means of RSVP hello
or some
fast IGP hellos mechanism on layer 2 link providing fast failure
notification.

However, when a link failure does occur, using the currently
deployed
mechanisms, a node adjacent to the failed link cannot tell
within the

time appropriate for Fast Reroute whether the node on the other side of that link is operational or not. Therefore, it is currently impossible to reliably tell apart the second and the third cases above. Hence, to protect both traffic that terminates at the failed node in case the failure was a link failure, and at the same time to protect traffic transit through the failed node in case it was a node failure, the LSR adjacent to the failed link is forced to use both the NHOP and the NNHOP tunnels at the same time. This may lead to a violation of bandwidth guarantees, since the NHOP and NNHOP tunnels were computed independently using the same backup bandwidth pool, and so they may share a link with enough bandwidth for only one but not the other.

A similar issue occurs in the case of bi-directional link failure. Since the two nodes on each side of the link will see the failure of an adjacent link, unless they can detect that it was a link and not a node failure, they will be forced to activate the NHOP tunnel protecting the link, and the NNHOP tunnel protecting the node on the other side. Essentially, the system will operate as if two failures have occurred simultaneously when in reality only a single (bi-directional) link failed.

This clearly can result in a violation of a bandwidth guarantee.

To address this issue, one needs a mechanism to differentiate a link from a node failure. Such a mechanism is described in [LINKNODE-FAILURE].

Note that in the centralized model, the server may compensate for the lack of the ability to tell a link from a node failure by making sure that the NNHOP bypass tunnels for adjacent nodes and the NHOP bypass tunnels for the corresponding links do not collide. While this makes the

problem of finding such backup tunnels algorithmically more challenging, it remains possible to achieve bandwidth sharing in this case. However, the ability to tell a link from a node failure is crucial for the distributed model when node protection is desired.

It is worth mentioning however that if just NHOP bypass tunnels are required (nodes are considered as reliable ''enough'') and just links are protected against failures, then there is no need to distinguish between node and link failure even in the distributed case.

8. Operations with links belonging to multiple SRLGs

In section 6 we limit the study to the case of links that are not part of more than one SRLG. However in some networks links might be part of more than one SRLG. This section presents the use of the facility based computation model in the general case where links are part of zero, one or more SRLGs. Both centralized and distributed scenarios are addressed.

Recall that facility based computation model consists of a coordinated placement of the set of bypass protecting one element by the same PCS, independently of the protection of each other element. This is clearly not applicable when bypass tunnels protect multiple independent elements, which is the case when bypass tunnels protect links belonging to multiple SRLGs, as an SRLG can be considered as an independent element (in terms of failure risk).

In case SRLGs are not disjoint, the placement of bypass LSPs protecting a given SRLG cannot be done independently of any other SRLG. Even if SRLGs remain independent elements in term of failure risk, their bandwidth protection computation can no longer be done independently, and must be coordinated.

For instance, lets take 3 links L1, L2, L3 and two SRLGs S1 and S2 such that S1= {L1, L2} and S2={L2, L3}. S1 and S2 are not disjoint, and their intersection is the link L2. If b1, b2 and b3 are NHOP bypass tunnels protecting respectively L1, L2, and L3 then:

- b1 and b2 computations must be coordinated, as they protect a common SRLG S1.
- b2 and b3 computations must be coordinated as they protect a common SRLG S2.

It results clearly that b1, b2 and b3 path computations must be coordinated, (and thus in the framework of facility-based computation model must be performed by the same PCS) and we say that L1, L2 and L3 are SRLG dependant.
It is important to note in this case that even if b1 and b3 protect independent elements, in terms of failure (L1 and L3 are SRLG diverse), their path computation must be coordinated.

Bandwidth sharing can still be ensured in that case, but this additional level of dependency in the computation of bypass LSPs requires more

intelligence on the server, and can substantially reduce the degree of distribution in case of a distributed setting.

The use of the facility based computation model, in this context, requires accounting for such dependency. The proposed solution is to regroup together all links whose protection placement must be coordinated into a new entity called Shared SRLG Dependency Link Group (SDLG). These links are said SRLG dependant. The result of such grouping is a set of disjoint groups, called Shared SRLG Dependency Link Groups, and noted SDLG.

Then, in the context of the facility based computation model, we extend the notion of facility to SDLGs. Each SDLG is treated, as a single element and is protected as a single entity (as a link or node), but with a modified aggregate bandwidth constraints, in order to take into account the assumption that only one SRLG fails and thus that not all bypass tunnels protecting a given SDLG are activated simultaneously.

This is discussed in more detail below.

8.1. Notion of SRLG dependency, and Shared SRLG Dependency Link Group (SDLG)

To take into account, in the facility based computation model, links that take part of multiple SRLGs, we define the notion of SRLG dependency: two links are said SRLG dependant, in the context of the facility based computation model, if their protection cannot be computed independently, or in other words if the computation of the NHOP bypass tunnels protecting these links must be done in a coordinated manner.

It is clear that if two links are part of the same SRLG then they are SRLG dependant, but this is not necessary. Two SRLG diverse links maybe

SRLG dependant, indeed in the above example, L1 and L3 are SRLG diverse but SRLG dependant.

Note that this dependency relation is transitive. It means that if L1 and L2 are dependant and L2 and L3 are dependant then L1 and L3 are dependant.

We define a Shared SRLG Dependency Link Group, noted SDLG, as a group of SRLG dependant links. An SDLG regroups all links that are SRLG dependant. From the transitivity property mentioned above, a link cannot belong to two SDLGs. Thus, it results that every link of a network, part of one ore more SRLGs, can be associated with a unique SDLG. The union of all the disjoint SDLGs is the set of links in the network.

The number of SDLGs will depend on the repartition of SRLGs among network links.

The number of SDLGs is always less than the number of SRLGs. At most (best case), nb SDLG = nb SRLG: this corresponds in fact to the particular case where all network links are part of 0 or one SRLG.

At least (worst case) nb SDLG =1: it is the case where all SRLGs are linked, i.e. we cannot find two disjoint SRLGs.

It is worth pointing out that a SDLG is no more than a union of linked SRLGs (ie a union of non disjoint SRLGs). An SDLG can be viewed as a union of SRLGs whose bandwidth protection computation must be done in a coordinated manner.

Thus a SDLG is noted S1 U S2 ... U Sk. This significantly simplifies the manipulation of SDLGs by LSRs, and the algorithm to determine the set of SDLGs.

The identification of SDLGs is required in a distributed computation. We propose to use as SDLG id, the lowest id of the union of SRLGs that compose the SDLG.

See Appendix E for an example.

8.2. SDLG protection

The key idea to support links that belong to multiple SRLGs, in the facility based computation model, is to treat an SDLG as a single element, and protect it as a single entity (as links or node). The placement of the set of bypass tunnels protecting links from an SDLG is performed independently of any other element.

The procedure is then relatively similar to the one for other elements (links or nodes). The computation of the set of tunnels protecting links of an SDLG, is performed in a coordinated manner, ignoring bandwidth of any bypass LSP protecting a distinct element (link, node or SDLG). The only distinction relies on the aggregate bandwidth constraint. Bypass tunnels computed for protection of an SDLG may protect different SRLGs. Thus, assuming than only one SRLG fails simultaneously, these bypass tunnels are not all activated simultaneously and it results that the aggregate bandwidth constraint on a link is lower than the cumulated bypass bandwidth. It is in fact the maximal bandwidth protecting an SRLG (see Appendix E for more details).

The PCS SHOULD take this specific aggregate bandwidth constraint into account when computing the placement of bypass tunnels corresponding to an SDLG to maximize the bandwidth sharing ratio.

It is clear that the problem it has to solve is algorithmically more challenging than the simple problem of the placement of given bandwidth demands on a network of given topology. Here the problem it has to solve is how to find a feasible placement for a set of NON-ALL-SIMULTANEOUS flows of given demands, in a network of given topology.

Both the centralized and distributed scenarios are supported. The centralized scenario requires no modification to what is defined in section 6.1, except the addition of the specific aggregate bandwidth constraint. By contrast, distributed computation requires a procedure specific to SDLGs that is specified in the section bellow.

8.2.1. Distributed scenario for SDLGs protection.

The same approach as defined in 6.2.3, is used to achieve a distributed SDLG protection. We propose that one of the end-nodes of the links forming the SDLG, be elected as PCS for whole SDLG. By default, the node with the lowest IGP id serves as PCS for the whole SDLG.

PLR processing:

- A PLR dynamically finds the SDLG its adjacent links belong to. (see Appendix E for a proposed algorithm to build SDLGs),
- Then it determines for each SDLG, the corresponding PCS (ie the end-node with the lowest IGP id), and sends a Path

computation request to these PCS, indicating the SDLG id (in the resource id field of the BACKUP-TUNNEL object).

Note 1: In the particular case where all links are part of zero or one SRLG, a SDLG is reduced to a single SRLG, and the resulting distributed setting is then identical to what is proposed in 6.2.3. Thus SDLG protection supports networks were links belong to 0 or one SRLG.

Note 2: In case all links are SRLG dependent, there is only one SDLG, and the result is a centralized computation (single PCS).

Note 3: As soon as there is one link in the network that belongs to multiple SRLGs, the SDLG approach must be used.

8.3. Alternative solution

An alternative solution to solve the problem of the computation of NHOP bypass tunnels protecting links part of multiple SRLGs could be to simply compute separate bypass LSP per SRLG for links belonging to multiple SRLGs. If the PLR could detect, upon the failure of a link, which of the SRLGs to which the link belongs actually failed, it could then use the appropriate bypass tunnel. In this case, each SRLG could be protected independently.

However, this approach clearly requires that a PLR is capable of determining which SRLG actually fails when it observes a failure of a link belonging to multiple SRLGs. Unfortunately, no mechanism to identify which of the SRLGs actually failed currently exists.

9. Operations with DS-TE and multiple Class-Types

This section assumes the reader is familiar with Diff-Serv-aware MPLS Traffic Engineering as specified in [DSTE-REQTS] and [DSTE-PROTO] and with its associated concepts such as Class-Types (CTs), Bandwidth Constraints (BCs) and the Russian Dolls bandwidth constraint model defined in [RDM].

The bandwidth protection approach described in this document supports DS-TE and operations with multiple Class-Types.

It is worth mentioning that both the primary and backup bandwidth pools sizes have to be carefully determined by the network administrator as their values dictate the congestion level in case of failure, as discussed bellow. In the absence of failure, up to the max reservable bandwidth pool (i.e the primary bandwidth pool) of (primary) traffic will be forwarded onto a link. In case of failure, potentially up to "Primary bandwidth pool" + "backup bandwidth pool" of traffic will be active on a link. Various scenarios as to what the backup bandwidth should be reserved for, are discussed in the following sections. The determination of their values compared to the link speed is a critical factor.

9.1. Single backup pool

Several bandwidth protection scenarios only require the use of a single backup pool.

First, when a single Class-Type is used (i.e. network which do not use Diff-Serv or use Diff-Serv but only enforce a single bandwidth constraint to all the TE tunnels), bandwidth protection can be achieved via a single backup bandwidth pool.

Second, when multiple Class-Types are used, a single backup pool can be used to provide bandwidth protection to LSPs from a single Class-Type CTc, which is the active CT with the highest index c, (in other words the active CT with the smallest Bandwidth Constraint), while LSPs from the other Class-Types do not get bandwidth protection.

Here is an example of such scenario. Let's consider the following network where:

- DS-TE and the Russian Dolls bandwidth constraint model are used
- two Class-Types (CTs) are used:
  - o CT1 is used for Voice Traffic
  - o CT0 is used for Data traffic

From a bandwidth protection perspective, let's assume that:

- Voice traffic (i.e. CT1 LSPs) requires Bandwidth Protection during failure
- Data traffic (i.e. CT0 LSPs) does not need Bandwidth Protection during failure.

Let's further assume that the network administrator has elected to use the notion of backup pool and specify bandwidth requirements for bypass tunnels based on the full bandwidth pool of primary tunnels (i.e. BC1) as configured towards the protected facility (as opposed to the amount of bandwidth currently used by the primary LSPs requiring bandwidth protection; see Appendix B for a detailed discussion).

Then, for every link the network administrator will configure:

- BC0, the Bandwidth Constraint on the aggregate across all primary LSPs (CT0+CT1)

- BC1, the Bandwidth Constraint for primary CT1 LSPs
- BCbu, the Bandwidth Constraint for the Backup CT1 LSPs

The bandwidth requirement of each backup LSP is configured based on the value of BC1 configured towards the facility it protects. In other words, the backup LSPs are only sized to protect voice traffic transiting via the protected facility.

Purely for illustration purposes, the diagram below builds on the one presented in section 9 of [DSTE-PROTO] to represent these bandwidth constraints in a pictorial manner.

```
                 I-------------------------------------------------------I
----------------I
                 I--------------I
I                 I
                 I    CT1       I
I                 I
                 I    Primary   I
I                 I
                 I--------------I                                       I   CT1
Backup     I
                 I                 CT1 + CT0
I                 I
                 I-------------------------------------------------------I
----------------I

                 I-----BC1------>
                 I---------------------------------------------BC0------> I----
BCbu------->
```

Note that while this scenario assumes Data traffic does not need Bandwidth protection during failure, Data traffic can be either not protected at all by Fast Reroute or be protected by Fast Reroute but without bandwidth protection during failure. In the former case, CT0 LSPs transporting Data traffic would not be rerouted into backup LSPs on failure. In the latter case, CT0 LSPs would be rerouted onto backup LSPs upon failure; the bypass tunnels could either be a different set of bypass tunnel from the bypass tunnels for voice, or could be the same bypass tunnels as for Voice assuming appropriate DiffServ marking and scheduling differentiation are configured properly, as discussed below.

From a scheduling perspective, a possible approach is for Voice traffic to be treated as the exact same Ordered Aggregate (i.e. use the same EF PHB) whether it is transported on primary LSPs or on backup LSPs. In that case, on a given link, BC1 and BCbu must clearly be configured in such a way that the Voice QoS objectives are met when there is simultaneously, on that link, up to BC1 worth of traffic on primary CT1

LSPs and up to BCbu worth of Voice Traffic on backup LSPs. A more detailed discussion on scheduling is provided in the following section.

The size of the backup pool BCbu is configured on all links such that the CT1 LSP QoS objectives are met when there is simultaneously, on that link, up to BC1 worth of primary LSPs and up to BCbu worth of backup CT1 traffic.

Notes
- If the objective for CT1 traffic is only to protect CT1 bandwidth then the network administrator must just make sure that: BC1+BCbu<Link Speed. If the objective is also to guarantee low jitter for CT1 traffic, one may desire to make sure that BC1+BCbu<U*Link Speed where U<1. Also as discussed bellow, the scheduling must be set appropriately.

- If BCbu+BC0>Link Speed, CT0 traffic may experiment congestion during failure but CT1 traffic is still bandwidth-protected.

Other scenarios can be addressed with a single bandwidth pool. This includes the case where all Class-Types need bandwidth protection but it is acceptable to relax delay guarantee to these classes during the failure and only offer bandwidth protection. Operations is very similar to the previous scenario described (e.g. size bypass tunnel based on BC0), the only difference is that QoS objectives other than bandwidth guarantee of other CTs than CT0 are not necessarily guaranteed to be preserved during failure. These CTs only get bandwidth assurances.

9.2. Multiple backup pools

When DS-TE is used and multiple Class-Types are supported, the operations described above can be easily extended to multiple bandwidth pools in the case where backup LSPs are sized based on the actual amount of established LSPs (See appendix B for discussion on the pros and cons of this approach): one backup pool can be used to separately constrain the bandwidth used by backup LSPs of each Class-Type.

In that case, each CT can be given bandwidth protection during failure with guarantee that each CT will also meet all its respective QoS objectives during the failure and without any bandwidth wastage.

Here is an example of such scenario. Let's consider the following network where:

- DS-TE and the Russian Dolls bandwidth constraint model are used
- two Class-Types (CTs) are used:
  - o CT1 is used for Voice Traffic
  - o CT0 is used for Data traffic

From a bandwidth protection perspective, let's assume that:
- Voice traffic (i.e. CT1 LSPs) needs Bandwidth Protection during failure

- Data traffic (i.e. CT0 LSPs) also needs Bandwidth Protection during failure.

Let's further assume that the network administrator has elected to specify bandwidth requirements for bypass tunnels based on the actual amount of established primary LSPs requiring bandwidth protection (as opposed to the full bandwidth pool of primary tunnels as configured towards the protected facility; see Appendix B for a detailed discussion).

Then, for every link the network administrator will configure:
- BC0, the Bandwidth Constraint on the aggregate across all primary LSPs (CT0+CT1)
- BC1, the Bandwidth Constraint for primary CT1 LSPs
- BCbu0, the Bandwidth Constraint on the aggregate across all backup LSPs (CT0+CT1)
- BCbu1, the Bandwidth Constraint on the CT1 backup LSPs

The bandwidth requirement of each CT0 backup LSP is configured based on the actual amount of established CT0 primary LSPs it protects. The bandwidth requirement of each CT1 backup LSP is configured based on the actual amount of established CT1 primary LSPs it protects.

Purely for illustration purposes, the diagram below represents these bandwidth constraints in a pictorial manner.

```
I----------------------------------------------
```

```
I-------------------I
                 I--------------I                              I----------
I         I
                 I    CT1       I                              I  CT1
I         I
                 I    Primary   I                              I  Backup
I         I
                 I--------------I                              I----------
I         I
                 I                CT1 + CT0 Primary            I   CT1+CT0
Backup   I
                 I----------------------------------------------
I-------------------I


                 I-----BC1------>                               I--BCbu1-->
                 I------------------------------------BC0------>I-------
BCbu0------->
```

The size of the backup pool BCbu0 is configured on all links such that the CT0 LSP QoS objectives are met when there is simultaneously, on that link, up to BC0 worth of CT0 primary LSPs and up to BCbu0 worth of backup CT0 traffic.

The size of the backup pool BCbu1 is configured on all links such that the CT1 LSP QoS objectives are met when there is simultaneously, on that link, up to BC1 worth of CT1 primary LSPs and up to BCbu1 worth of backup CT1 traffic.

In the case where backup LSPs are sized based on the amount of reservable bandwidth (See appendix B for discussion on the pros and

cons of this approach), it is also possible to extend operations to multiple bandwidth pools in the same way, but this may result in bandwidth wastage. This is because BC1 will be effectively

reserved
both from BC1bu and from BC0bu (with the RDM model).

Here is an example of such scenario. Let's consider the following network where:

- DS-TE and the Russian Dolls bandwidth constraint model are used
- two Class-Types (CTs) are used:
  - o CT1 is used for Voice Traffic
  - o CT0 is used for Data traffic

From a bandwidth protection perspective, let's assume that:

- Voice traffic (i.e. CT1 LSPs) needs Bandwidth Protection during failure
- Data traffic (i.e. CT0 LSPs) also needs Bandwidth Protection during failure.

Let's further assume that the network administrator has elected to specify bandwidth requirements for bypass tunnels based on the full bandwidth pool of primary tunnels as configured towards the protected facility (as opposed to the amount of bandwidth currently used by the primary LSPs; see Appendix B for a detailed discussion).

Then, for every link the network administrator will configure:

- BC0, the Bandwidth Constraint on the aggregate across all primary LSPs (CT0+CT1)
- BC1, the Bandwidth Constraint for primary CT1 LSPs
- BCbu0, the Bandwidth Constraint on the aggregate across all backup LSPs (CT0+CT1)
- BCbu1, the Bandwidth Constraint on the CT1 backup LSPs

The bandwidth requirement of each CT1 backup LSP is configured based on the value of BC1 configured towards the facility it protects. The bandwidth requirement of each CT0 backup LSP is configured based on the value of BC0 configured towards the facility it protects. Thus, effectively the CT1 backup LSP and CT0 backup LSP will have an aggregate bandwidth requirement of BC0+BC1 which represents a bandwidth wastage since we know that the aggregate primary bandwidth across CT0 and CT1 is actually limited to BC0 (since BC0 is a bandwidth constraint on CT0+CT1).

Operations with multiple backup pools will be discussed in more details in subsequent versions of this draft.

10. Interaction with scheduling

The bandwidth protection approach described in this document does not require any enhancement or modification to MPLS scheduling mechanisms

beyond those defined in [MPLS-DIFF]. In particular, scheduling can remain entirely unaware of Fast Reroute and bandwidth protection; in particular this approach does not require that scheduling behave differently depending on whether a packet is transported on a primary LSP or a backup LSP, nor does it require per-LSP scheduling.

This approach simply requires that the existing MPLS scheduling mechanisms (e.g. Diff-Serv PHBs) are configured in a manner which is compatible with the goal of bandwidth protection, because while the bandwidth protection allocates bandwidth appropriately in the control plane, it is scheduling which is responsible for the actual enforcement in the data path of the corresponding service rates to packets in a way which will achieve the targeted bandwidth protection.

The details of which configuration is appropriate depends on multiple parameters such as the details of the Diff-Serv policy, the

bandwidth
protection policy and the number of DS-TE Class-Types supported.
Thus,
it is outside the scope of this draft.

For illustration purposes, we can expand on the scheduling
aspects in
the example discussed in the previous section. A possible
scheduling
approach based on MPLS Diff-Serv is the following:
- let's assume Voice uses EF PHB and Data uses
AF11 ,AF12, AF21
and AF22 PHBs
- E-LSPs with preconfigured EXP<-->PHB mapping can be
used
with:
  o EXP=eee maps to EF
  o EXP=aa0 maps to AF11
  o EXP=aa1 maps to AF12
  o EXP=bb0 maps to AF21
  o EXP=bb1 maps to AF22
- separate E-LSPs are established for Voice and for Data
- Voice E-LSPs are established in CT1
- Data E-LSPs are established in CT0
- Separate E-LSPs are established for backup (voice and
data)
constrained by Bcbu (but with signaled bandwidth set
to zero
as discussed in section 6).
- BC1 and BCbu are configured on every link so that the
EF PHB
can guarantee appropriate QoS to voice when there is
BC1+BCbu
worth of voice traffic
- The uniform Diff-Serv tunneling mode defined in
section 2.6
of [MPLS-DIFF] is used on the bypass tunnels. In
particular,
when a packet is steered into a bypass tunnel by the
PLR
(i.e. when the bypass tunnel label entry is pushed
onto the
packet) the EXP field of the packet is copied into
the EXP
field of the bypass tunnel label entry.

Then, upon a failure:
- voice packets have their EXP=eee regardless of whether
they
are transported on a primary tunnel or bypass tunnel.
Thus

they will be scheduled by the EF PHB. Since our bandwidth protection approach ensures that there is less CT1 LSPs than

BC1 and less CT1 backup LSPs than BCbu, and since we have configured BC1 and BCbu so that EF can cope with that aggregate load, QoS is indeed guaranteed to voice during failure.

- Data packets have their EXP=aax or EXP=bbx regardless of whether they are transported on a primary tunnel or a bypass tunnel. Thus, it is clear that they do not steal bandwidth from the EF PHB.

In the example described in the previous section, we mentioned several possible protection policies for Data. Let's assume that Data is protected by Fast Reroute but without Bandwidth protection and let's assume that the same bypass tunnels are used as for voice. Then it must be noted that even if Data is injecting traffic into the backup LSPs (whose bandwidth constraint do NOT factor such load since they only factor the voice traffic), this does NOT compromise the voice bandwidth protection in anyway since:

- the admission control performed over backup LSPs factored the voice load over the EF PHB
- the data packets transported on the backup LSP have their EXP=aax or EXP=bbx and thus are scheduled in the AF PHBs without affecting the EF PHB.

On the other hand, Data packets may experience QoS degradation during failure. This is because a given link, in addition to data packets on primary CT0 LSPs for which admission control has been performed, may also receive data packets on backup LSPs for which effectively no admission control has been performed (since this load was not factored in the sizing of the backup LSPs). This is in line with the assumption that Data traffic did not need bandwidth protection during failure.

In the particular case where the PLR could not establish a bypass tunnel with the full requested amount of bandwidth (due to some lack of bandwidth in the backup pool) and instead established a bypass tunnel with a smaller bandwidth, when rerouting LSPs onto this bypass tunnel, the PLR may ensure that the amount of rerouted primary LSPs complies with the actual bandwidth of the bypass tunnel. This can done using the same bypass tunnel (or a separate bypass tunnel) with the pipe DiffServ tunneling mode for the non bandwidth protected primary rerouted TE LSPs (this both includes the set of TE LSPs not requiring bandwidth protection and the set of TE LSP that have required bandwidth protection but for which there was not enough backup bandwidth on the bypass tunnel to accommodate their request). Otherwise, this would simply violate bandwidth protection (for traffic on this bypass tunnel as well as for all traffic on any LSP using the same PHBs) because more traffic than reserved for would end up in the bypass tunnel.

11. Routing and signaling extensions

11.1. Routing (IGP-TE) extensions

In this section, we define an IGP-TE routing extensions to signal the bandwidth protection pool. This extension is identical to the extension defined in [BP-PLACEMENT] and is defined for ISIS-TE and OSPF-TE.

As explained earlier, this extension is purely optional and can be considered as useful but not mandatory.

One new sub TLVs (in Link TLVs of TE LSA for OSPF, and in IS reachability TLVs for ISIS) is defined:

backup bandwidth pool sub-TLV: this sub-TLV contains the maximum backup bandwidth that can be reserved on this link in this direction (from the node originating the LSA to its neighbors). The backup bandwidth is encoded in 32 bits in IEEE floating-point format. The units are bytes per second.

OSPF and ISIS types are TBD.

The format of the TLVs within the body of a Router Information LSA is the same as the TLV format used by the Traffic Engineering Extensions to OSPF [OSPF-TE].

```
     0                   1                   2
3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+
    |              TBD               |
4                |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+
    |                     backup
bandwidth              |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+
```

OSPF Backup bandwidth pool sub-TLV

The IS-IS backup bandwidth pool sub-TLV just differs from the format depicted above by the code type and length fields that are 1 byte long.

Again, the bandwidth protection pool being announced by IGP-TE is a static value i.e does not dynamically change as backup TE LSP are set up, which preserves IGP scalability.

As the bandwidth protection pool is being defined on a per link basis, this allows for different policies depending on the link characteristics.

Note that the format might change in the future to support multiple backup bandwidth pools.

11.2. Signaling (RSVP-TE) extensions

11.2.1. PCC -> PCS signaling : specification of a set of constraints

The PCC (an LSR) will provide to the PCS a set of constraints to satisfy for the bypass tunnel path computation. The PCC-PCS signaling protocol used in this draft is based on [PATH-COMP]. A new object called BACKUP-TUNNEL, related to bypass tunnel is defined in this section.

As defined in [PATH-COMP], the path computation request has the following format:

```
<Path computation request> ::=  <Common Header> [ <INTEGRITY> ]
                                   [ <MESSAGE_ID_ACK> |
                                   <MESSAGE_ID_NACK>] ... ]
```

[<MESSAGE_ID>]
<SESSION>
<REQUEST_ID>
[<NB_PATH>]
[<EXPLICIT_ROUTE>]
[<METRIC_TYPE>]
[<EXCLUDE_ELEMENT>]
[<BACKUP-TUNNEL>]
[<SESSION_ATTRIBUTE>]
[<POLICY_DATA> ...]
<sender descriptor>

<sender descriptor> ::= <SENDER_TEMPLATE> <SENDER_TSPEC>
[<ADSPEC>]
[<RECORD_ROUTE>]

There are several constraints that should be taken into account when computing the bypass tunnel paths that have already been described in section 6.3:

- element to protect,
- bandwidth,
- affinities,
- Max number of bypass tunnels, (per link or per pair of links through a node)
- Minimum bandwidth on a single bypass tunnel,
- CT to protect,
- Existing bypass tunnels,
- other optional parameters, e.g. maximum allowed propagation delay increase of the bypass tunnel over the segment of the primary path protected by the tunnel.

Some are optional (see bellow).

The PCC can make use of a single path computation request even if multiple bypass tunnel path computations are requested. In that case, the PCC must include a separate BACKUP-TUNNEL object per request. For instance, if multiple NHOP bypass tunnels path computations are requested, the PCC could send a unique RSVP path computation request to the PCC with one BACKUP-TUNNEL per each bypass tunnel path to be computed.

BACKUP-TUNNEL Class-Num is [TBD by IANA] - C-Type is [TBD by IANA]

```
 0                   1
2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8
9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|     Flags     |    Reserved     |     ETP       |
CT      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|                        Resource-
ID                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|                    Bypass-tunnel-
destination                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|
Bandwidth                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|                        Include-
any                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|                        Exclude-
any                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|                        Include-
all                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
|                     MAX-NB-BACKUP-
TUNNEL                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
+-+-+-+
```

| MIN-BW-BACKUP-
TUNNEL |
+-
+-+-+-+

Flags: 8 bits

0x01: specifies that the requesting PCC provides a set (possibly reduced to a single element) of existing bypass tunnels. For each existing bypass tunnel the corresponding ERO will be included within the Path computation request.

0x02: specifies to the PCS that in case of negative reply (the PCC cannot find a set of bypass tunnels that fulfill the set of requirements), the PCS should provide in the path computation reply the best possible set of bypass tunnels i.e the set of bypass tunnels that will protect the maximum possible amount of bandwidth for the protected element.

0x04 (G bit): as mentioned earlier, the PCC might decide to protect either a bandwidth pool or the sum of the actual reserved bandwidths by the set of TE LSPs requiring bandwidth protection. In the first case (called a global bandwidth protection request,

the G bit must be set), the PCC just needs to specify the ETP, CT and Ressouce-ID fields and optionally the bandwidth. The Bypass-tunnel-destination field must be set to 0.
In the second case (the G bit must be cleared), the required

amount of protected bandwidth per NNHOP must also be specified. So for each NNHOP, a separate BACKUP-TUNNEL object must be included in the path computation request sent to the PCS, with the bypass tunnel destination address and required bandwidth.

0x08: when set, this bit indicates that the PCC cannot differentiate link from node failure. This should be taken into account by the PCS when computing NNHOP backup tunnels to avoid collision of NNHOP backup tunnels from adjacent nodes (see section 7). This bit must be cleared if the PCC can differentiate a link from a node failure. This bit must be cleared for link, SRLG or SDLG protection.

ETP (Element to protect): 8 bits
0x00: Link
0x01: Node
0x02: SRLG
0x03: SDLG

CT: Class-type to protect

Resource ID: identifies the resource to protect
- for a link, the PCC must specify the link IP address,
- for a node, the PCC must specify one of the interface IP addresses of the node or its router ID,
- for a SRLG, the PCC must specify the SRLG number
- for a SDLG, the PCC must specify the SDLG id (which is the lowest SRLG id)

Bypass-tunnel-destination

Bandwidth: (32-bit IEEE floating point integer) in bytes-per-second.

Affinities (optional)

This parameter is optional and must be set to 0x00000000 if not used.

Exclude-any

A 32-bit vector representing a set of attribute filters associated with a backup path any of which renders a link unacceptable.

Include-any

A 32-bit vector representing a set of attribute filters Associated with a backup path any of which renders a link acceptable (with respect to this test). A null set (all bits set to zero)automatically passes.

Include-all

A 32-bit vector representing a set of attribute filters Associated with a backup path all of which must be present for a link to be acceptable (with respect to this test). A null set (all bits set to zero) automatically passes.

MAX-NB-BACKUP-TUNNEL: Maximum number of bypass tunnels

This parameter is optional and must be set to 0x00000000 if not used.

MIN-BW-BACKUP-TUNNEL: Minimum bandwidth of any element of the backup tunnel set.

This parameter is optional and must be set to 0x00000000 if not used.

11.2.2. PCS -> PCC signaling - sending the computed set of bypass tunnels

After having processed a PCC request, the PCS will send a path computation reply to the PCC.

The likelihood of finding a solution that will obey the set of constraints will of course be conditioned by:

- the network resources (and particularly the backup bandwidth/link bandwidth ratio)
- the set of constraints.

There are two possible results:

- the request can be satisfied (positive reply)
- the new request cannot be (fully) satisfied (negative reply).

As defined in PATH-COMP, the PCS' path computation reply message will have the following form:

```
<Path Computation Reply>::=<Common Header> [ <INTEGRITY> ]
                [<MESSAGE_ID_ACK> |   <MESSAGE_ID_NACK>]...]
                [ <MESSAGE_ID> ]
                <REQUEST_ID>
                [ <NB_PATH> ]
                [<BACKUP-TUNNEL> <EXPLICIT_ROUTE> [<LSP-BANDWIDTH>]
                [<PATH_COST>]] ...
```



```
                [ <ERROR_SPEC>]
                [<NO_PATH_AVAILABLE] ]
                [ <POLICY_DATA> ... ]
```

For each BACKUP-TUNNEL object present in the path computation request, the Path Computation Reply will contain:

- A BACKUP-TUNNEL object specifying the characteristics of the computed bypass tunnel(s) (identification of the resource it protects (ETP, resource-ID, ...),
- Followed by the path(s) of the computed bypass tunnel(s) (EXPLICIT_ROUTE) and their respective computed bandwidth (if

different from the respective request).

A set of bypass tunnels may be reduced to a single element if the PCS can find a single bypass tunnel that fulfills the requirements.

11.2.3. Examples

Consider the following network:

```
                R4
                 /
                /
     R1------R2------R3
                \
                 \
                  R5
```

Example 1:

- Backup bandwidth requirement is based on the max reservable primary bandwidth,
- R1 (PCC) sends a request to R2 (PCS) for a set of CT1 bypass tunnels to guard against a failure of R2, with a bandwidth requirement of 50M.
- The result must contain a maximum of 5 bypass tunnels per NNHOP, with a minimum bandwidth 5M for each bypass tunnel,
- In case of negative reply, the server should provide the best possible set of tunnels

This is a global bandwidth protection request.

Request:
<SESSION>
<REQUEST-ID>=a
<BACKUP-TUNNEL>: flag: G=1, ETP=0x01, CT=0x01
resource-id= R2 address
Bypass-tunnel-destination=0x00000000
bandwidth=50M
min-bw=5M
Max-tunnel=5

other fields set to 0x00000000

The reply is positive, the result is a set of 6 paths:
For NNHOP R4, there are two bypass, b1 (bw 30M) and b2
(bw 20M)
For NNHOP R3, there are three bypass, b3 (bw 30M), b4
(bw 10M),
b5 (bw10M)
For NNHOP R5, there is one bypass, b6 (50M)

```
Reply:
<Request-ID>=a
<NB-PATH>: number-path=6
<BACKUP-TUNNEL>: flag: G=1, ETP=0x01, CT=0x01
resource-id= R2 address
bandwidth=50M
other fields set to 0x00000000
<ERO b1> <LSP-BANDWIDTH>: bw =30M
<ERO b2> <LSP-BANDWIDTH>: bw =20M
<ERO b3> <LSP-BANDWIDTH>: bw =30M
<ERO b4> <LSP-BANDWIDTH>: bw =10M
<ERO b5> <LSP-BANDWIDTH>: bw =10M
<ERO b6> <LSP-BANDWIDTH>: bw =50M
```

Example 2:

- Backup bandwidth requirement is based on the current reserved primary
bandwidth
- R1 sends a request to R2 for a set of CT1 bypass tunnel to protect R2,
with a bandwidth requirement for NNHOPs R3 and R4 :
R3=10M
R4=20M
- The result must contain a maximum of 5 bypass LSPs per NNHOP, with a
minimum bandwidth 1M
- In case of negative reply, the server should provide the best possible
set of tunnels

```
Request:
<SESSION>
<REQUEST-ID>=a
<BACKUP-TUNNEL>: flag: G=0, ETP=0x01, CT=0x01
resource-id= R2 address
```

Bypass-tunnel-destination=R3 address
bandwidth=10M
min-bw=1M
Max-tunnel=5
<BACKUP-TUNNEL>: flag: G=0, ETP=0x01, CT=0x01
resource-id= R2 address
Bypass-tunnel-destination=R4 address
bandwidth=20M
min-bw=1M
Max-tunnel=5

The reply is negative, the best solution found by the PCS R2 is:
For NNHOP R3, the best solution is 9M, with two bypass, b1 (bw 6M) and b2 (bw 3M)
For NNHOP R4, the best solution is 15M , with two bypass b3 (10M) and b4(5M)

Reply :

<REQUEST-ID>=a
<ERROR-SPEC>
<NO-PATH-AVAILABLE>: flag: G=0, constraint-type=0x0001,
<NB-PATH>: num-path=4
<BACKUP-TUNNEL>: flag=0x02, ETP=0x01, CT=0x01
resource-id= R2 address
<ERO b1> <LSP-BANDWIDTH> bw=6M,
<ERO b2>, <LSP-BANDWIDTH> bw=3M
<ERO b3> <LSP-BANDWIDTH> bw=10M,
<ERO b4>, <LSP-BANDWIDTH> bw=5M

12. Bypass tunnel - Make before break

In case of bypass tunnel path change, the new bypass tunnel may be set up using make before break. This may or not be possible depending on the change in the set of bypass tunnels.

13. Stateless versus Statefull PCS

There are basically two options for the PCS:
- can be statefull: the PCS registers the various bypass tunnels computation requests and results. It will also monitor the network states (bypass tunnels in place, ...)
- can be stateless: the PCS does not maintain any state. This approach is the recommended approach for the distributed model.

14. Packing algorithm

Once the set of bypass tunnels is in place and their respective bandwidth, the PLR should, for each protected TE LSP successfully signaled, select a corresponding bypass tunnel. As per defined in [FAST-REROUTE], the bandwidth protection requirement for the protected LSP can be specified using the FAST-REROUTE object or by setting the ''Bandwidth protection desired'' bit in the SESSION-ATTRIBUTE of the Path message. Based on the signaled backup bandwidth requirement for the protected LSP, the PLR should appropriately select the bypass tunnel to use for the protected TE LSP, making sure the requested backup bandwidth requirement is met.

15. Interoperability in a mixed environment

There could potentially be some interoperability issues when conformant and non conformant nodes to this draft are mixed within the same network. The following interoperability issues categories could be identified:

* Ability of LSRs to communicate with the server: if the PCS is an LSR,

other LSRs need to communicate with the server using the signaling extensions proposed in this draft,

* Interaction of different bandwidth protection FRR techniques.
- networks not supporting backup bandwidth pools,
- interaction with bypass tunnels using explicit bandwidth reservation,

Interoperability issues will be covered in detailed in a further revision of this draft.

16. Security Considerations

The practice described in this draft does not raise specific security issues beyond those of existing TE.

17. Acknowledgements

The authors would like to thank Carol Iturralde, Rog Goguen, Vishal Sharma, Shahram Davari and Renaud Moignard for their useful comments.

18. Intellectual Property

CISCO SYSTEMS represents that it has disclosed the existence of any proprietary or intellectual property rights in the contribution that are reasonably and personally known to the contributor. The contributor does not represent that he personally knows of all potentially pertinent proprietary and intellectual property rights owned or claimed by the organization he represents (if any) or third parties.

References

[TE-REQ] Awduche et al, Requirements for Traffic Engineering over MPLS, RFC2702, September 1999.

[OSPF-TE] Katz, Yeung, Traffic Engineering Extensions to OSPF, draft-katz-yeung-ospf-traffic-05.txt, June 2001.

[ISIS-TE] Smit, Li, IS-IS extensions for Traffic Engineering, draft-
ietf-isis-traffic-03.txt, June 2001.

[RSVP-TE] Awduche et al, "RSVP-TE: Extensions to RSVP for LSP Tunnels",
RFC3209, December 2001.

[CR-LDP] Jamoussi et al., "Constraint-Based LSP Setup using LDP",
draft-ietf-mpls-cr-ldp-05.txt, February 2001

[METRICS] Fedyk et al, ''Multiple Metrics for Traffic Engineering with
IS-IS and OSPF'', draft-fedyk-isis-ospf-te-metrics-01.txt, November
2000.

[DS-TE] Le Faucheur et al, ''Requirements for support of Diff-Serv-aware
MPLS Traffic Engineering'', draft-ietf-tewg-diff-te-reqts-06.txt,
September 2002.

[PATH-COMP] Vasseur et al, ''RSVP Path computation request and reply
messages'', draft-vasseur-mpls-computation-rsvp-03.txt, November 2002.

[FAST-REROUTE] Pan, P. et al., "Fast Reroute Techniques in
RSVP-TE", Internet Draft, draft-ietf-mpls-rsvp-lsp-fastreroute-02.txt
, February 2003

[BP-PLACEMENT] Leroux, Calvignac, ''A method for an Optimized Online
Placement of MPLS Bypass Tunnels'', draft-leroux-mpls-bypass-placement-
00.txt, February 2002.

[KINI] Kini et al, ''Shared Backup Label Switched Path

Restoration'',
draft-kini-restoration-shared-backup-01.txt, May 2001.

[MPLS-DIFF] RFC3270, Le Faucheur et al, " Multi-Protocol Label Switching
(MPLS) Support of Differentiated Services'', May 2002.

[RDM] Le Faucheur, ''Russian Dolls Bandwidth Constraints Model for
Diff-Serv-aware MPLS Traffic Engineering'', draft-ietf-tewg-diff-te-
russian-01.txt, February 2003.

[IGP-CAP] Aggarwal et al, ''Extensions to IS-IS and OSPF for Advertising
Optional Router Capabilities'', Internet draft, draft-raggarwa-igp-cap-
01.txt, October 2002.

[OSPF-TE-TLV] Vasseur, Psenak ''Traffic Engineering capability TLV for
OSPF'', Internet draft, work in progress.

[LINKNODE-FAILURE] Vasseur, Charny, ''Distinguish a link from a node
failure using RSVP Hellos extensions'', draft-vasseur-mpls-linknode-
failure-00.txt, work in progress.

[RFC3469] Sharma V., et al, "Framework for Multi-Protocol Label
Switching (MPLS)-based Recovery", Feb, 2003

[INTER-AS-TE-REQS] Zhang et al, "MPLS Inter-AS Traffic Engineering
requirements", draft-zhang-interas-te-req-01.txt (work in progress).

[INTER-AS-TE] Vasseur and Zhang, "Inter-AS MPLS Traffic Engineering",
draft-vasseur-inter-as-te-00.txt, February 2003 (work in progress)

Authors' Address:

Jean Philippe Vasseur
Cisco Systems, Inc.
300 Apollo Drive
Chelmsford, MA 01824
USA
Email: jpv@cisco.com

Anna Charny
Cisco Systems, Inc.
300 Apollo Drive
Chelmsford, MA 01824
USA
Email: acharny@cisco.com

Francois Le Faucheur
Cisco Systems, Inc.
Village d'Entreprise Green Side - Batiment T3
400, Avenue de Roumanille
06410 Biot-Sophia Antipolis
France
Phone: +33 4 97 23 26 19
Email: flefauch@cisco.com

Javier Achirica
Telef nica Data Espa±a
Beatriz de Bobadilla, 14
28040 Madrid
Spain
javier.achirica@telefonica-data.com

Jean-Louis Le Roux
France Telecom
2, avenue Pierre-Marzin
22307 Lannion Cedex
France
E-mail: jeanlouis.leroux@francetelecom.com

Appendix A: Limitations/inefficiency of the independent CSPF-based computation model

Let's give a simple illustration of the case where PLRs are using an independent based CSPF approach and fail to find a feasible placement of the bypass tunnels. In this case we assume that no load-balancing of the backup tunnels is allowed. Note that similar (although more complicated) examples could be provided for a given (bounded) number of load-balanced tunnels protecting the same element.

```
            R6---------R7
             |\         |
             | \        |
             |  \       |
      R1----R2---R3----R4----R5
             |          |
             |          |
             |          |
            R8---------R9
```

The goal is to find the bypass tunnels protecting node R3.

Let's assume that the amount of bandwidth than needs to be protected on links adjacent to R3 is given by:

R6-R3=5M
R2-R3=10M

Assume further that bandwidth on other links available for placement of the bypass tunnels is as follows:

R6-R7=10M
R6-R2=20M
R2-R8=5M
other links=100M

Bandwidth on a link in each direction is assumed the same (e.g. link R8-R2 is also 5M).

In a distributed and non coordinated setting, the order in which the direct neighbors of R3 compute and place their bypass tunnels protecting against the failure of R3 can be arbitrary.

Suppose R6 tries to compute a NNHOP bypass tunnel to R4 with bandwidth 5M and selects the shortest path to R4 with available bandwidth and bypassing R3. That is R6-R7-R4. When R2 tries to compute a NNHOP bypass tunnel to R4 with bandwidth 10M, it discovers that there in no feasible path it can take. In contrast, and independent server using a more sophisticated algorithm could discover this condition and find that the solution:

NNHOP bypass tunnel from R6 to R4: R6-R2-R8-R9-R4 (BW=5M),
NNHOP bypass tunnel from R2 to R4: R2-R6-R7-R4 (BW=10M),
NNHOP bypass tunnel from R4 to R2: R4-R7-R6-R2 (BW=5M),
NNHOP bypass tunnel from R4 to R6: R4-R9-R8-R2-R6 (BW=10M),
NNHOP bypass tunnel from R6 to R2: R6-R2 (BW=5M),
NNHOP bypass tunnel from R2 to R6: R2-R6 (BW=5M)

satisfies the constraints. Since the general problem of finding a feasible placement of given bandwidth demands in a general-topology network is well-known to be NP-complete, it could be argued that a centralized server cannot be expected to implement an algorithm that is always guaranteed to find a solution in a reasonable time in all cases anyway. While it is certainly true, it is quite clear that a server-based implementation can run a heuristic algorithm that is much more likely to find a solution than simple greedy CSPF-based approach. Moreover, the centralized model is much more amenable to supporting various optimality criteria not available with the simple CSPF-based approach.

Appendix B: Bandwidth to protect

There are two different approaches for the bandwidth constraint of the bypass tunnels.

The bypass tunnel bandwidth may be based on:
- the amount of reservable bandwidth on a particular network resource,
- the sum of bandwidths actually reserved by established TE LSPs requesting bandwidth protection on a particular resource.

Solution 1: primary reservable pool

In this case, the bypass tunnel bandwidth requirement is based on the primary reservable pool we need to protect.

Example:

```
      R6---R7----R8
       |\   |   / |
       | -- | --  |
       |   \|/    |
R1----R2---R3----R4----R5
       |   / \    |
       | --   --  |
       |/       \ |
      R9---------R10
```

Objective: find a set of bypass tunnels from R2 to R4 to protect R2 from a node failure of R3.

In this case, the bypass tunnel bandwidth requirement is being driven by the smaller of amount of max reservable bandwidth (the bandwidth pools) defined on the links R2-R3 and R3-R4 (potentially multiplied by some factor), independently on the current state of bandwidth reservation on these links. In case of nested pools of bandwidth, the outmost pool could be taken into account (that would cover all pools nested inside) or just one of the subpools.

With this solution 1, in the example above, when R2 requests the server to compute for it the bypass tunnels protecting its traffic traversing R3 against R3's failure, it should request the computation of 6 different NNHOP bypass tunnels with headend in R2 and tailend at each other direct neighbor of R3. The bandwidth of each of these bypass tunnels is determined by the minimum of the max reservable bandwidth of the pool for which protection is desired on the link R2-R3 and the link connecting R3 to the corresponding neighbor. For example, if max reservable bandwidth is 10 Mbps on link R2-R3, and 8 Mbps on link R3-R4, then the bypass tunnel from R2 to R4 must have the bandwidth of 8Mbps available to it.

The obvious benefit of this approach is of course that the backup path computation is not impacted by the dynamic network state (the TE LSPs currently in place) which is a serious advantage in term of stability. A new backup path computation should just be triggered in case of network topology change (link/node down, change in the reservable amount of bandwidth on a given link, ...). The drawback is that the bandwidth requirement may be substantially higher than needed if the actual amount of capacity is much larger than the actual amount of reserved capacity of the TE LSPs in place; the higher is the bandwidth requirement for the bypass tunnel, the lower is the likelihood to find a solution.

Aggregate bandwidth constraints for bypass tunnels

When protecting a bi-directional link, an SRLG, a SDLG or a node, multiple bypass tunnels are typically required. For example, a bi-directional link protection requires at least one bypass tunnel for each of the two directions of the link. For SRLG, at least one (or two in the bi-directional case) bypass tunnel is required for each link in the SRLG. For SDLG, at least one (or two in the bi-directional case) bypass tunnels are required for each link of the SDLG. For a node, at least one bypass tunnel is required for every pair of direct neighbors of this node.

At first glance, it may seem that if tunnels T1,T2,...TK with bandwidth requirements b1,b2,..Bk protecting against a failure of some element F traverse some link L, then link L must have at least b1+b2+...+bk bandwidth available for backup placement. It is indeed always true for link and SRLG protection.

For SDLG protection, link L must have at least max(bw (SRLGi)) bandwidth available for backup placement (see Appendix E). A path computation server should take such aggregate constraint into consideration when computing bypass tunnel placement.

For node protection, when the actual amount of primary bandwidth is protected, the above statement is also true. However, for the case when the backup pool is protected, this statement is unnecessarily conservative.

To see this, consider the above example, and assume that the primary pools (max reservable bandwidth for a particular subpool) on all links adjacent to R3 are 10 Mbps, except for the link R3-R4, which has the primary pool of 8 Mbps in each direction. Note now that bypass tunnels T1 (R6-R4) and T2(R2-R4) each need 8 Mbps. However, the total amount of primary traffic traversing paths R6-R3-R4 and R2-R3-R4 is bounded by the primary pool of link R3-R4, and so the aggregate bandwidth requirements of both backups tunnels is only 8Mbps, and not 16Mbps. A path computation server implementing solution 1 SHOULD take such aggregate constraints into consideration when computing bypass tunnels placement.

Solution 2: total amount of bandwidth actually reserved on a given link

Another option is to make the bypass tunnel bandwidth requirement a function of the actual amount of reserved bandwidth for the set of TE LSPs requesting bandwidth protection. In the diagram above, R2 would request a set of bypass tunnels so that the backup bandwidth is equal to the sum of the bandwidths of the currently established TE LSPs crossing the R2-R3 link. This value may be multiplied by some factor to allocate some spare room for new coming TE LSPs.

With this solution, R2 would send a request to the PCS for the actual amount of reserved bandwidth between it and each of the direct neighbors of R3 to which it has primary traffic. For example, if there is no primary TE LSP established between R2 and R6, there is no need to request a bypass tunnel connecting R2 to R6. Furthermore, if the total bandwidth of all TE LSPs between R2 and R4 traversing R3 is 2 Mbps, then the bandwidth requirement of the bypass tunnel R2-R4 can be 2 Mbps instead of 8Mbps in solution 1.

Note however, that the bypass tunnels are signaled with zero

bandwidth
and therefore do not reserve any bandwidth. Therefore, as long as the
set of bypass tunnels protecting the entire pool exist (and can be
found by the algorithm computing their placement), the bandwidth
savings of solution 2 over solution 1 is irrelevant. However in the
cases when the backup bandwidth is so scarce that the bypass tunnels
protecting the entire bandwidth pools cannot be found, solution 2
clearly provides a benefit.

The main drawback of solution 2 is the need for a potentially large
number of bypass tunnel recomputations each time TE LSPs are set
up/torn down which creates additional load on the device computing the
placement, and results in additional signaling overhead. Furthermore,
recomputing and resignaling the new set of bypass tunnels may take some
(albeit relatively short) time, leaving all primary TE LSPs traversing
the affected elements temporarily unprotected.

The risk of instability may be reduced by the use of some UP/ DOWN
thresholds. In this case, each time a new TE LSP is set up, if a UP
threshold is crossed a new bypass tunnel path computation is triggered.
Optionally, a DOWN threshold scheme may be used to better optimize the
backup bandwidth usage. In this case, when a TE LSP is torn down, if a
DOWN threshold is crossed, a bypass tunnel path computation is
triggered. For obvious reasons, it is expected to have different UP and
DOWN thresholds.

Mix of solutions 1 and 2: another approach is also to combine the two
solutions described above.

Suppose the objective of full bandwidth protection cannot be met by the
PCS: in case of negative reply from the PCS that cannot find a solution

to the requested constraints, some algorithms may be implemented to find the best possible solution (the closest to the initial request).

Three options exist:
- option 1: the intelligence is on the PCC. The PCC will send several requests to the PCS until it gets a positive reply.
- option 2: the intelligence is on the PCS. The PCS in case of negative reply tries to find the ''best'' possible solution and suggests those new values to the PCC. Then the PCC will decide whether it can accept the new values. If yes, it will resend a new request to the PCS with the suggested value to get the result. Option 2 requires less signaling overhead than option 1.
- option 3: the PCS directly answers with the best possible solution.

Option 3 requires less signaling overhead than option 2.

1) in solution 1 all bandwidth information is available at the PCS, so there is actually no need to signal the bandwidth at all

2) in solution 2 or a mix, the server may or may not have primary bandwidth info (e.g. is an LSR ''protects itself'', it already knows all the actual primary bandwidth requirements, but if a PCS protects some other element, in this case primary bandwidth needs to be communicated to it.

Appendix C: Bypass tunnel path computation triggering and path changes

This appendix deals with:
- bypass tunnel path computation triggers,
- bypass tunnel path changes,

Bypass tunnel path computation triggers will of course depends on whether solution 1 or 2 has been adopted (see Appendix B).

With solution 1: primary reservable pool

Bypass tunnel path computation may be triggered when the network resource to protect first comes up or when the first protected LSP is signaled.

This is a matter of local policy.

Then the bypass tunnel path computation is triggered:

- when the network topology has changed. Following a
network
failure (link/node), the PLR may decide, after some
configurable time has elapsed, to trigger a new path
computation. This includes the situation where a new
neighbor
of an already protected node comes up. This is a
topology
change.
- when the reservable bandwidth of the protected section
changes,
- when the amount of bandwidth protection pool changes,
- when a bypass tunnel path reoptimization is triggered:
a PCC
may desire to trigger a bypass tunnel path computation
at any
time (using for instance a timer driven approach) in
order to
see whether a more optimal set of bypass tunnels could
be
found.
- note that it might be desirable to trigger bypass
tunnel
computation at regular intervals (send a new bypass
tunnel
computation when a timer expires). The periodic bypass
tunnel
computation is expected to happen at a low frequency.

With solution 2: sum of the bandwidth actually reserved on a
given link

Bypass tunnel path computation is triggered:
- when the network topology has changed. Following a
network
failure (link/node), the PLR may decide, after some
configurable time has elapsed, to trigger a new path
computation. This includes the situation where a new
neighbor
of an already protected node comes up. This is a
topology
change.
- when the reservable bandwidth of the protected section
changes,
- when the amount of bandwidth protection pool changes,
- when the actual amount of reserved bandwidth changes
(e.g
when a TE LSP is setup or torn down, or when a UP/DOWN
threshold is crossed)

all,

- when a bypass tunnel path reoptimization is triggered: a PCC may desire to trigger a bypass tunnel path computation at any time (using for instance a timer driven approach) in order to see whether a more optimal set of bypass tunnels could be found.

Bypass tunnel path changes

Various conditions may generate some changes of existing bypass tunnels paths:

(1) when a bypass tunnel path computation has been triggered and as a result a new set of bypass tunnels has been computed that differs from the already in place setup (because the bypass tunnel constraints have changed or a more optimal bypass tunnel path exists),

(2) when as a result of a new backup path computation that has been triggered by another node, the PCS has computed a new set of bypass tunnels for the node.

(1) is obvious.

Example of (2)

```
      R6---R7----R8
       |\   |   / |
       | -- | --  |
       |   \|/    |
R1----R2---R3----R4----R5
       |   / \    |
       | --   --  |
       |/       \ |
      R9---------R10
```

As an example, suppose:

- Max backup bandwidth pool size along the R6-R7-R8-R4 path is 10M

- Max backup bandwidth pool size along the R2-R9-R10-R4 path is 15M

- On R6, the bypass tunnel T1 to protect R6-R3-R4:
Min(R6-R3,R3-R4)=10M
Bypass tunnel T1: path=R6-R7-R8-R4, bandwidth=10M

- On R2, the bypass tunnel T2 to protect R2-R3-R4:
Min(R2-R3,R3-R4)=5M
Bypass tunnel T2: path=R2-R9-R10-R4, bandwidth=5M

For some reason, R6 triggers a new bypass tunnel path computation, requesting for more bandwidth (15M).

To satisfy this new constraint, the PCS will find the following solutions:

T1: R6-R2-R9-R10-R4
T2: R2-R6-R7-R8-R4

Which implies to reroute T2, although the backup requirements of R2 have not changed.

This example shows that a change in a set of bypass tunnels for a node may have some consequences on the set of bypass tunnels for some other nodes.

Appendix D PLR State machine

As discussed in Appendix C, a bypass tunnel request from a node X may result in some changes of the set of bypass tunnels for other nodes.

In this case, upon the receipt of a bypass tunnel path computation request, the PCS needs to trigger a simultaneous computation of bypass tunnels for all its neighbors and, in turn, needs to return the sets of bypass tunnels to all its neighbors (this includes not only the requesting node but also all the PCS' neighbors).

The corresponding finite state machine would be:

(1) When a new bypass tunnel path computation is triggered (see appendix C), the PCC sends a request to the PCS specifying a set of constraints (see section 6.3).

(2) When receiving a bypass tunnel path computation request, the PCS will:
(2.1) Optionally first request the set of bandwidth requirements and bypass tunnels already in place to all its neighbors. See note 2 bellow.
(2.2) Perform the bypass tunnel path computation simultaneously for all its neighbors.
Two different situations may happen:
(2.2.1) the new request cannot be (fully) satisfied. In this case, as defined in [PATH-COMP], the PCS will send a negative reply including a NO-PATH-AVAILABLE object. Optionally, this object may indicate the constraint that could not be fulfilled and also optionally a suggested value for this constraint for which a solution could have been found. The PCS may use an algorithm to find the closest solution to initial request. Optionally, as previously discussed, the PCS may return the closest possible solution that could be found.
(2.2.2) the new request can be satisfied.
(2.3) send the new sets of bypass tunnel to each neighbor
(2.4) each PCS' neighbor will then compare the new set of bypass tunnel(s) to the already in place set of bypass tunnels. In case of no change, then stop. If the new set of bypass tunnel differs from the set of bypass tunnels already in place, the node will tear down the existing bypass tunnels and sets up the new set of bypass tunnels optionally with a make before break (if possible).

Note 1: if a PCC request cannot be fully satisfied by the PCS, as discussed above, some algorithm may be used to find the closest possible solution to the request. In this case, the PCS will provide the set of bypass tunnels and the amount of protected bandwidth. This means the node will be partially protected (i.e the amount of protected bandwidth is less than the amount of setup TE LSPs/reservable bandwidth).

Note 2: this may be a very beneficial optimization if the PCS is capable of minimizing the incremental change. A statefull PCS will have the knowledge of the existing bypass tunnels. A stateless PCS will have, upon the receipt of the bypass tunnel path computation request, to poll its neighbors to get the sets of existing bypass tunnels as well as the other parameters (this would imply some additional signaling extension to [PATH-COMP]).

Appendix E: Procedure with Shared SRLG Dependency link Groups (SDLG)

As defined in section 8, SDLGs regroup all links whose backup computation must be coordinated. Each SDLG is a union of SRLGs and is identified by the lowest SRLG id.

Two SRLGs are said ''linked'' if there is a least one link that belongs to both of them (in other words if they are not disjoints).

A simple algorithm can be found to determine the set of SDLGs. In the centralized scenario, the algorithm is run only by the central PCS. In the distributed scenario, the algorithm is run by each LSR, but limited to the determination of SDLGs its protected adjacent links belong to.

Example (taken from an operational network)

```
R8----R3-----R4----R6
|    / |    / | \   |
|   /  |   /  |  \  |
|  /   |  /   |   \ |
R1-----R2----R5----R7
```

List of SRLGs

SRLG 1 = {R1-R2, R2-R3}
SRLG 2 = {R2-R5, R2-R4}
SRLG 3 = {R2-R5, R4-R5}
SRLG 4 = {R2-R4, R4-R5}
SRLG 5 = {R4-R6, R4-R7}
SRLG 6 = {R1-R3, R3-R8}

The above algorithm allows to rapidly determine SDLGs :

There are four SDLGs in this network:
SDLG 1 = SRLG 1 = {R1-R2, R2-R3}
SDLG 2 = SRLG 2 U SRLG 3 U SRLG 4 = {R2-R5, R2-R4, R4-R5}
SDLG 5 = SRLG 5 = {R4-R6, R4-R7}
SDLG 6 = SRLG 6 = {R3-R8, R1-R3}

SDLG id = min (SRLG id)

In a distributed scenario, if we assume the following IGP id order

R5 < R4 < R8 < R1 < R2 < R7 < R6 < R3, then:
-R1 is elected as PCS for SDLG 1
-R5 is elected as PCS for SDLG 2
-R4 is elected as PCS for SDLG 5
-R8 is elected as PCS for SDLG 6

Distribution degree

We define the distribution degree (DD) of a distributed facility based computation scenario, as the of number of PCS(es) used divided by the

number of elements to protect.
Examples:
-Full distribution: DD = 1
-Central server : DD = 1/number of elements to protect

The degree of distributed computation in case of SDLG will depend directly on the number of SDLGs, that depends itself on the repartition of SRLGs among network links.
The distribution efficiency can be expressed as:
DD= nb (SDLG) / nb (links belonging to one or more SRLGs)

In the above example DD= 0.4

Aggregate bandwidth constraint for bypass tunnels of the same SDLG

Bypass tunnels computed for protection of an SDLG may protect different SRLGs. Thus, assuming than only one SRLG fails simultaneously, these bypass tunnels are not all activated simultaneously and it results that the aggregate bandwidth constraint is lower than the cumulated bandwidth.

If tunnels T1,T2,...,Tk with bandwidth b1,...,bk protecting links from SDLG S that is the union of SRLG 1,...,L, traverse some link L, then, the aggregate bandwidth constraint on L is
B= Max (bw (SRLG i)) where bw (SRLG i) = Sum (bj, Tj protecting SRLG i).

L must have at least B bandwidth available for backup placement.

Example:

In the above figure, in case of SDLG 2 protection, if bypass tunnels T1 (50M), T2 (30M) and T3 (20M), protecting respectively links R2-R5, R2-R4 and R4-R5, traverse the same link L, then the aggregate bandwidth constraint is not 100M but 80M (max (sum(30+50), sum (20+30)), as only two of them can be activated simultaneously, under the single failure assumption.

The problem of the placement of a given bandwidth demand based on this collision criteria is often called "Non Simultaneous Multi Commodity Flow Problem" in the literature, it is well know to be NP-COMPLETE. Heuristics to solve this problem are algorithmically more complex than the one used to solve the classical problem of the placement of a set of flows of given demand in a network of given topology (used in case the element to protect is a simple link or node).